THIS OPINION WAS NOT WRITTEN FOR PUBLICATION

The opinion in support of the decision being entered today was **not** written for publication and is **not** binding precedent of the Board.

Paper No. 21

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte NOBUKI TOMINAGA,AKIRA TAWAKA,
and SEIICHI URUSHIBARA

Appeal No. 2001-0712
Application 08/148,887[1]

ON BRIEF

Before THOMAS, FLEMING and GROSS, **Administrative Patent Judges**.

FLEMING, ***Administrative Patent Judge***.

DECISION ON APPEAL

[1] Application for patent filed November 8, 1993.

This is a decision on appeal from the final rejection of claims 11-21, all of the claims pending in the present application. Claims 1-10 have been cancelled.

The invention relates to a resource allocation device (figure 1, number 33; figure 2) for use by a compiler to reduce run time and size of a machine language program (specification, page 3, lines 14-25). The resource allocation device includes an allocation generation unit (figure 2, number 1), an expression tree generation unit (figure 2, number 2), a template (figure 2, number 3), an instruction selection unit (figure 2, number 4), a cost table (figure 2, number 5), a cost detection unit (figure 2, number 6), a total cost computation unit (figure 2, number 7), a best pattern decision unit (figure 2, number 8), a cost estimation unit (figure 2, number 9), a variable judging unit (figure 2, number 10), a selection operation decision unit (figure 2, number 11), and a variable storage (figure 2, number 12).

The allocation pattern generation unit generates all conceivable patterns of variables and resources (specification, page 13, lines 10-12). The expression tree generation unit generates an expression tree for each of the operations included in the program portion extracted by the optimization device (specification, page 13, lines 23-25). The template shows a correspondence between detection items and corresponding instruction sequences (specification, page 15, lines 1-2).

The instruction selection unit examines a specific expression tree generated by the expression tree generation unit and a specific allocation pattern generated by the allocation pattern generation unit, and then picks the instruction sequence from the template that corresponds to that expression tree and that generated allocation pattern. This unit then uses one of the allocation patterns to determine, for that allocation pattern, the resources to which the variables for the operand and the operation result storage are allocated. From the template this unit finds the instruction sequence corresponding to the action type, variable type, operand type, and the resources dictated by the allocation pattern (specification, page 15, lines 23-26 through page 16, lines 1-18).

The cost table shows instruction sequences and the number of execution clock cycles required for execution of each instruction sequence (specification, page 17, lines 3-6). The cost detection unit detects the number of execution clock cycles for each of the instruction sequences extracted by the instruction sequence selection unit by referring to the cost table (specification, page 17, lines 7-10). The total cost computation unit figures out total cost of each allocation pattern generated by the allocation pattern generation unit by summing the number of clock cycles detected by the cost detection unit (specification, page 17, lines 11-14). The best allocation pattern detection unit detects the allocation pattern with the lowest total cost in all of the allocation patterns generated by the pattern generation unit (specification, page 17, lines 15-17).

Independent claim 11 is reproduced as follows:

> 11. A resource allocation device for use by a compiler, the resource allocation device for translating a high-level language program or an intermediate language program into a machine language of a target machine, the resource allocation device performing resource allocation by allocating variables of a number of operations included in a program portion of the program to resources such as registers and memories, the resource allocation device comprising:
>
> variable holding means for holding the variables included in the program portion;
>
> allocation pattern generation means for generating allocation patterns, each allocation pattern defining a different allocation of the variables to the resources;
>
> instruction sequence holding means for holding instruction sequences for different combinations of the operations and the resources, each instruction sequence corresponding to one of the operations and written in an assembly language and/or a macro language;
>
> instruction sequence extraction means for extracting

from the instruction sequence holding means instruction sequences, a number of instruction sequences equal to the number of operations being extracted for each allocation pattern, the instruction sequence extracting means generating, for each allocation pattern, a program which corresponds to the program portion and which comprises extracted instruction sequences;

a cost table for holding both the extracted instruction sequences and a corresponding cost for each extracted instruction sequence, each cost representing a number of execution clock cycles required by execution of the corresponding extracted instruction sequence;

cost detection means for detecting, from the cost table, the cost of each extracted instruction sequence;

total cost computation means for computing a total cost of each allocation pattern by adding the costs of the number of extracted instruction sequences for the allocation pattern; and

best pattern determining means for determining an allocation pattern with a lowest cost by referring to the total cost of each allocation pattern, the resource allocation thereby being performed in accordance with the determined allocation pattern.

The Examiner relies on the following references:

Charles Y. Hitchcock III et al. (Hitchcock), “A Method of Automatic Data Path Synthesis,” pp. 484-89, IEEE (1983).

Chi-Hung Chi et al. (Chi), “Register Allocation for GaAs Computer Systems,” System Sciences, Vol. I, pp. 266-274, IEEE/IEE (1988).

Claims 11-21 stand rejected under 35 U.S.C. § 103 as being unpatentable over Hitchcock in view of Chi.

Rather than reiterate the arguments of Appellants and the Examiner, reference is made to the Brief[2], the Reply

[2] The Brief was received March 6, 1996.

Brief[3] and the Examiner's Answer[4] for the respective details thereof.

OPINION

We will not sustain the rejections of claims 11-21 under 35 U.S.C. § 103.

The Examiner has failed to set forth a ***prima facie*** case. It is the burden of the Examiner to establish why one having ordinary skill in the art would have been led to the claimed invention by the express teachings or suggestions found

in the prior art, or by implications contained in such teachings or suggestions. ***In re Sernaker***, 702 F.2d 989, 995, 217 USPQ 1, 6 (Fed. Cir. 1983). The Federal Circuit states that “[t]he mere fact that the prior art may be modified in the manner suggested by Examiner does not make the modification obvious unless the prior art suggested the

[3] The Reply Brief was received July 19, 1996.

[4] The Examiner's Answer was mailed May 17, 1996. A letter in response to the Reply Brief was mailed February 6, 2001 and it stated that the reply brief had been entered and considered but no further response by the Examiner was necessary.

desirability of the modification.”

In re Fritch, 972 F.2d 1260, 1266 n.14, 23 USPQ2d 1780, 1783-84 n. 14 (Fed. Cir. 1992), *citing* ***In re Gordon***, 733 F.2d 900, 902, 221 USPQ 1125, 1127 (Fed. Cir. 1984). The Federal Circuit reasons in ***Para-Ordnance Mfg. v. SGS Importers Int’l Inc.***,

73 F.3d 1085, 1088-89, 37 USPQ2d 1237, 1239-40 (Fed. Cir. 1995), that for the determination of obviousness, the court must answer whether one of ordinary skill in the art who sets out to solve the problem and who had before him in his workshop the prior art, would have reasonably expected to use the solution that is claimed by Appellants. However, “[o]bviousness may not be established using hindsight or in view of the teachings or suggestions of the invention.” ***Para-Ordnance Mfg. v. SGS Importers Int’l***, 73 F.3d at 1087, 37 USPQ2d at 1239, ***citing***

W. L. Gore & Assocs., Inc. v. Garlock, Inc. 721 F.2d 1540, 1551, 1553, 220 USPQ 303, 311, 312-13 (Fed. Cir. 1983), **cert. denied**, 469 U.S. 851 (1984). In addition, our reviewing court requires the PTO to make specific findings on a suggestion to combine prior art references. ***In re Dembiczak***, 175 F.3d 994, 1000-01, 50 USPQ2d 1614, 1617-19 (Fed. Cir. 1999).

On pages 11-13 of the brief, Appellants argue that there is no suggestion in the prior art to combine the teachings of Hitchcock with the teachings of Chi. Specifically, Appellants point out that Hitchcock teaches[5] a method of automatically synthesizing data paths from a behavioral description of a hardware design. The EMUCS program implementing the method attempts to find a minimum cost implementation of a hardware design given a dataflow representation VT (Appellants' emphasis). The program proceeds by binding abstract data flow elements onto hardware elements in iterative fashion. Once all the hardware elements have been bound, the program determines which implementation would result in the optimum design given the parameter which is to be optimized.

Appellants assert[6] that Chi teaches[7] a graph-based

[5] Page 484, column 1.

[6] Brief, page 12.

[7] Page 286, column 1; page 270, columns 1-2 and table 2.

scheme for allocating variables to registers and memory, and this scheme is based on a machine state level model (MSL). This model is directed to optimizing register storage based on analyzing read and write operations for register and memory locations.

It is then pointed out by Appellants that neither reference teaches or suggests the desirability of using the teachings of the other to produce Appellants' invention, and that Hitchcock is not directed to developing a resource allocation device for a compiler, but for an automated hardware design tool. Furthermore, there is no teaching concerning the need or desirability of extracting instruction sequences to determine an optimized compiler output.

Finally in this regard, Appellants contend that the reasons to combine as set forth by the Examiner[8] are not directed to the combination of references supporting the rejection, and fails to consider the effects that the actual instruction sequences have on performance.

[8] Final rejection, page 3, last three lines through page 4, lines 1-5.

It is then argued by Appellants[9] that contrary to the Examiner's assertion, Hitchcock's disclosed abstract flow elements and binding means are not, respectively, the claimed variable holding means and allocation pattern generation means. The abstract flow elements of Hitchcock are noted to be hardware data flow requirements of the intended hardware design, while the variable holding means of the present invention stores the software variables which will be allocated to the system resources.

As to the binding means, Appellants assert that these means of Hitchcock bind abstract data flow elements onto hardware elements in a step-by-step fashion, and take a hardware description and attempt to create an optimum hardware design. The allocation pattern generation of the invention, however, generates an allocation pattern for every possible combination of software variables and computer resources.

Appellants then disagree[10] with the Examiner's statement that Chi discloses the remaining claim limitations.

[9] Brief, page 13.

[10] Brief, page 14.

They point out that in Chi the total of the read/write costs of each variable is used as a basis to assign variables to register locations, and only considering read/write costs may have a detrimental effect on overall execution speed if the operations to be performed are not considered. Thus, there is no teaching that the actual instruction sequences associated with the variables should be considered. Appellants assert that the claimed system overcomes this problem.

As to the instruction sequence extraction means of claim 11 and the instruction set generating means of claim 21, Appellants aver that they are not disclosed by the cited references, and cannot, as the Examiner contends, be inferred as necessary in order to determine the instruction sequence for computing cost. Appellants note that neither reference teaches the need or desire to evaluate the actual instruction sequences associated with the variables to be allocated. Hitchcock is noted to lack any reference to extracting the actual instruction sequences associated with a particular variable allocation in order to determine the cost of the overall computer program. Chi is noted not to extract actual instructions associated with a variable allocation, and not to consider the effect that the operation associated with a given variable has on a register allocation.

Finally, Appellants argue[11] that the final rejection was applied generally without specifically showing that the limitations of the dependant claims are disclosed by the cited prior art.

In the answer[12], the Examiner asserts that there is incentive to combine the references and states "Considering Hitchcock et al as the primary reference, one of ordinary skill in the art would modify Hitchcock in order to obtain an optimum allocation as taught by Chi. Considering Chi as the primary reference, one of ordinary skill in the art would be motivated to extract and try all possible variable-register/memory combinations in order to automatically determine all possible bindings and hence obtain a complete cost profile".

As regards Appellants' argument that there is no teaching concerning the need or desirability of extracting

[11] Brief, pages 20-22.

[12] Pages 3 and 4.

instruction sequences to determine an optimized compiler output, the Examiner points to Chi's[13] recitation of live variable ranges, asserting that Chi only considers the instruction sequences within a specific block.

In addition, the Examiner states[14] that "The office action does **not only** address allocating all possible variables to

[13] Pages 268 and 269.

[14] Answer, page 5.

registers. Chi reference also meets limitations of the claims in particular with respect to the execution costs."

In regard to Appellants' argument that hardware data flow elements and software variables are not analogous, especially in terms of computing cost, the Examiner asserts that it is unclear whether Appellants consider data flow elements a hardware item, and that Hitchcock shows allocating variables to registers. The Examiner also notes that Chi is all about allocation of variables of a program to registers.

In addressing Appellants' contention that Hitchcock does not disclose allocating variables to registers in the same fashion as the invention, the Examiner points to Hitchcock's teaching[15] of finding all possible bindings, and the cost calculations.

As regards Appellants' arguments directed to considering only read/write costs versus instruction sequence processing the Examiner asserts[16] that the "reversal phenomenon" does not exist and/or is not applicable here.

[15] Page 485.

[16] Answer, page 8.

Regarding Appellants' contention that neither reference discloses the claimed instruction sequence holding means of claim 11 or the holding means of claim 16, the Examiner asserts[17] that it is well known that assembly language programs have all the data which will actually be allocated to registers/memory, and that the cited references clearly illustrate this fact.

Finally, as regards the limitations in the dependent claims, the Examiner applies teachings of Chi or Hitchcock to each limitation.

As pointed out by our reviewing court, we must first determine the scope of the claim. "[T]he name of the game is the claim.” ***In re Hiniker Co.***, 150 F.3d 1362, 1369, 47 USPQ2d 1523, 1529 (Fed. Cir. 1998).

Turning first to Appellants' claim 11, we note that the second subparagraph of this claim recites, "allocation pattern generation means for generating allocation patterns, each allocation pattern defining a different allocation of the variables to the resources". Upon a careful review of

[17] Answer, page 9.

Hitchcock, we find that contrary to the Examiner's assertion, the means binding all abstract elements to the pieces of hardware fails to meet this claim limitation. It is noted all of the appealed independent claims require such allocation pattern generating means.

We also agree with Appellants that Chi does not disclose the limitations of subparagraphs 3 and 4 of claim 11. In Chi, the total of the read/write costs of each variable is used as a basis to assign variables to register locations, and only considering read/write costs can have a detrimental effect on overall execution speed if the operations to be performed are not considered. Thus, there is no teaching that the actual instruction sequences associated with the variables should be considered.

The instruction sequence extraction means of claim 11 and the instruction set generating means of claim 21 are not disclosed by the cited references, and cannot, as the Examiner contends, be inferred as necessary in order to determine the instruction sequence for computing cost. Neither Hitchcock nor Chi teaches the need or desire to evaluate the actual instruction sequences associated with the variables to be allocated. Hitchcock lacks any reference to extracting the actual instruction sequences associated with a particular variable allocation in order to determine the cost of the overall computer program. Chi does not extract actual instructions associated with a variable allocation, and does not consider the effect that the operation associated with a given variable has on a register allocation.

Furthermore, we find that the very different subjects of the Hitchcock and Chi articles indicate against one skilled in the art combining their teachings. Hitchcock teaches[18] a method of automatically synthesizing data paths from a behavioral description of a hardware design. The EMUCS program implementing the method attempts to find a minimum cost implementation of a hardware design given a dataflow representation VT. The program proceeds by binding abstract data flow elements onto hardware elements in iterative fashion. Once all the hardware elements have been bound, the program determines which implementation would result in the optimum design given the parameter which is to be optimized.

Chi teaches[19] a graph-based scheme for allocating variables to registers and memory, and this scheme is based on a machine state level model. This model is directed to optimizing register storage based on analyzing read and write

[18] Page 484, column 1.

[19] Page 286, column 1; page 270, columns 1-2 and table 2.

operations for register and memory locations.

As Hitchcock is not directed to developing a resource allocation device for a compiler, but for an automated hardware design tool, and since neither reference teaches or suggests the desirability of using the teachings of the other to produce Appellants' invention, we find the Examiner's reasons to combine the teachings of these references to be inadequate.

As we noted above, the Federal Circuit states that "[t]he mere fact that the prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggested the desirability of the modification." ***In re Fritch***, 972 F.2d at 1266 n.14, 23 USPQ2d at 1783-84 n.14, **citing In re Gordon**, 733 F.2d at 902, 221 USPQ at 1127. "Obviousness may not be established using hindsight or in view of the teachings or suggestions of the inventor." ***Para-Ordnance***, 73 F.3d at 1087, 37 USPQ2d at 1239, ***citing W. L. Gore & Assocs.***, 721 F.2d at 1551, 1553, 220 USPQ at 311, 312-13. In addition, our reviewing court requires the PTO to make specific findings on a suggestion to combine prior art references. ***In re Dembiczak,*** 175 F.3d at 1000-01, 50 USPQ2d at 1617-19.

We are not inclined to dispense with proof by evidence when the proposition at issue is not supported by a teaching in a prior art reference or shown to be common knowledge of unquestionable demonstration. Our reviewing court requires this evidence in order to establish a ***prima facie*** case.

In re Piasecki, 745 F.2d 1468, 1471-72, 223 USPQ 785, 787-88 (Fed. Cir. 1984); ***In re Knapp-Monarch Co.***, 296 F.2d 230, 232, 132 USPQ 6, 8 (CCPA 1961); ***In re Cofer***, 354 F.2d 664, 668, 148 USPQ 268, 271-72 (CCPA 1966). Furthermore, our reviewing court states in ***In re Piasecki***, 745 F.2d at 1472, 223 USPQ at

788 the following:

> The Supreme Court in *Graham v. John Deere Co.*, 383 U.S. 1 (1966), focused on the procedural and evidentiary processes in reaching a conclusion under Section 103. As adapted to ex parte procedure, Graham is interpreted as continuing to place the "burden of proof on the Patent Office which requires it to produce the factual basis for its rejection of an application under section 102 and 103". *Citing In re Warner*, 379 F.2d 1011, 1016, 154 USPQ 173, 177 (CCPA 1967).

Therefore, we will not sustain the rejections of claims 11-21 under 35 U.S.C. § 103 as being unpatentable over Hitchcock in view of Chi.

Accordingly, the Examiner's decision is reversed.

REVERSED

JAMES D. THOMAS)
Administrative Patent Judge)
)
)
) BOARD OF
PATENT
MICHAEL R. FLEMING) APPEALS AND
Administrative Patent Judge)
INTERFERENCES
)
)
)
ANITA PELLMAN GROSS)
Administrative Patent Judge)

MRF/dal

Price, Gess & Ubell
2100 S.E. Main Street
Suite 250
Irvine, CA 92714